

Ai-Powered Attacks and Mitigation Strategies for Enterprises: A Comprehensive Research Review

Sarathkumar K¹, Pandian R², Shyamala Gowri B³, Karthiga R⁴

¹Principal Lead, Cybersecurity, Optiv Security Inc., Bangalore, Karnataka, India

²Research Scholar, Dept. of Computer Science, NITTTR, Chennai, Tamil Nadu, India

³Professor, Dept. of CSE, Saveetha School of Engineering, Saveetha University, Chennai, Tamil Nadu, India

⁴PG Scholar, Dept. of Cybersecurity, University of Madras, Chennai, Tamil Nadu, India

Emails: sharathsmart3@gmail.com¹, pandyaa.1306@gmail.com², shyamalagowribalaraman@gmail.com³, karthikarvkmr@gmail.com⁴

Abstract

The rapid adoption of artificial intelligence (AI), large language models (LLMs), retrieval-augmented generation (RAG), and autonomous AI agents is transforming enterprise information systems, cybersecurity operations, software development, customer services, and decision-making. However, the integration of AI introduces attack surfaces that extend beyond conventional application and network vulnerabilities. Attackers can exploit identities, application programming interfaces, prompts, documents, retrieval stores, vector databases, model artifacts, agent tools, browser sessions, software pipelines, cloud credentials, and AI gateways. This research review develops a 24-path enterprise attack taxonomy from the supplied reference model and organizes the attacks across identity and access, input and context, data and retrieval, model and supply chain, agent and workflow, and infrastructure and output domains. It reviews major attack classes and proposes a defense-in-depth mitigation architecture combining zero-trust identity, least privilege, data provenance, model integrity verification, secure tool execution, continuous monitoring, adversarial testing, human oversight, and AI-specific incident response. The paper also identifies research gaps involving prompt-injection detection, agent authorization, persistent memory, model provenance, AI supply-chain assurance, and enterprise-scale security evaluation.

Keywords: Artificial Intelligence, Enterprise Cybersecurity, Generative AI, LLM, Prompt Injection, RAG Poisoning.

1. Introduction

Artificial intelligence has moved rapidly from experimental research environments into mainstream enterprise computing. Organizations increasingly use machine learning and generative AI for software development, document processing, fraud detection, customer support, business intelligence, security operations, knowledge management, decision support, and autonomous workflow execution. Large language models have accelerated this transition because natural-language interfaces allow employees and applications to interact with enterprise information and services without traditional graphical interfaces or manually constructed queries.

(Autio et al., 2024; Jin et al., 2024; Zhang et al., 2024). (Wei et al., 2023; Carlini et al., 2021). This transformation also changes the cybersecurity problem. Traditional enterprise security primarily protects users, endpoints, networks, applications, databases, and cloud infrastructure. AI-enabled systems add foundation models, prompts, embeddings, vector databases, retrieval pipelines, model registries, agent memories, tool connectors, orchestration layers, inference APIs, and model gateways. Consequently, compromise may occur without exploitation of conventional software vulnerability; an attacker may manipulate trusted

information, influence model behavior, compromise an external model or package, or exploit authority granted to an autonomous agent. The supplied reference image illustrates this broader attack surface through 24 hidden attack paths in AI-powered enterprises. The paths include identity compromise, OAuth abuse, privilege creep, prompt injection, malicious documents, email-to-agent attacks, RAG poisoning, vector exposure, API-key theft, backend pivoting, tool-call abuse, MCP compromise, memory poisoning, model supply-chain compromise, data poisoning, model backdoors, output injection, IAM escalation, agent identity abuse, browser hijacking, AI-to-CI/CD attacks, repository leakage, cloud credential theft, and AI gateway compromise. The central argument of this review is that enterprise AI security cannot be achieved by adding a single prompt filter or content-moderation layer. Security must instead be implemented across the entire AI lifecycle and the surrounding enterprise environment. The paper therefore treats AI as a socio-technical system in which identities, data, models, tools, applications, and human decisions form a connected security boundary. The main objectives are to consolidate the 24 attack paths into a research-oriented taxonomy, relate each attack to practical enterprise controls, present a defense-in-depth architecture, and identify research directions for resilient enterprise AI.

2. Review Methodology

This paper uses a structured narrative review approach. The analysis combines established AI-risk and cybersecurity frameworks with academic studies on adversarial machine learning, large-language-model security, AI agents, poisoning, model backdoors, and supply-chain vulnerabilities. The primary framework sources include the NIST AI Risk Management Framework and its Generative AI Profile, the NIST Secure Software Development Framework community profile for generative AI and dual-use foundation models, the OWASP Top 10 for LLM Applications, and MITRE ATLAS (NIST, 2023; Autio et al., 2024; Booth et al., 2024; OWASP, 2025; MITRE, 2026). The supplied 24-path image was used as the conceptual starting point for the threat taxonomy rather than as a scholarly source. The duplicated number in the source image for

Repository Leakage is corrected to 19 so that the taxonomy contains exactly 24 distinct attack paths. The review then maps these paths to enterprise security layers and mitigation controls.

3. Enterprise Ai Threat Landscape

The attack surface of AI-enabled enterprises is broader than that of conventional enterprise applications. An AI system can simultaneously process user instructions, internal documents, web content, emails, database records, retrieved knowledge, and tool outputs. A malicious item embedded in any of these sources can influence subsequent model behavior. NIST emphasizes that generative AI introduces risks involving confabulation, data privacy, harmful bias, information integrity, and security, requiring risk management across the AI lifecycle (Autio et al., 2024). The fundamental security-boundary problem is that natural-language content can act both as data and as an instruction. This makes indirect prompt injection particularly significant in enterprise settings. When an agent can invoke tools, the impact can extend from misinformation to unauthorized transactions, data disclosure, record modification, or infrastructure changes. Agent-focused research demonstrates that prompt injection should be evaluated in realistic dynamic environments rather than only with static prompt datasets (Debenedetti et al., 2024). (Perez & Ribeiro, 2022; Shayegani et al., 2023). A useful enterprise attack chain is: external attacker → enterprise data → AI model → agent reasoning → tool invocation → enterprise resource. The compromise therefore does not necessarily require direct access to the protected resource. AI changes the meaning of privilege because an assistant may combine a user identity with access to email, cloud storage, databases, browsers, code repositories, and enterprise APIs.

4. Taxonomy Of 24 Ai-Powered Enterprise Attack Paths

Table 1 summarizes the 24 attack paths represented in the supplied reference model. The taxonomy is intended to support threat modeling, security-control selection, and incident-response planning. As shown in Table 1 Taxonomy of AI-powered attack paths in enterprise environments.

Table 1 Taxonomy of AI-powered attack paths in enterprise environments

Attack Path	Typical Impact
1. Identity Compromise	Unauthorized access
2. OAuth Abuse	Account takeover
3. Privilege Creep	Excessive authorization
4. Prompt Injection	Behavioral manipulation
5. Malicious Documents	Context poisoning
6. Email-to-Agent	Unauthorized actions
7. RAG Poisoning	Manipulated responses
8. Vector Exposure	Data leakage
9. API Key Theft	Service compromise
10. Backend Pivot	Lateral movement
11. Tool-Call Abuse	Unauthorized execution
12. MCP Compromise	Trusted workflow compromise
13. Memory Poisoning	Long-term manipulation
14. Model Supply-Chain Attack	Malicious model deployment
15. Data Poisoning	Altered model behavior
16. Model Backdoor	Triggered malicious behavior
17. Output Injection	Downstream exploitation
18. IAM Escalation	Expanded privileges
19. Repository Leakage	Secret disclosure
20. Agent Identity Abuse	Unauthorized resources
21. Browser Hijacking	Web-system compromise
22. AI-to-CI/CD Pivot	Production compromise
23. Cloud Credential Theft	Cloud takeover
24. AI Gateway Compromise	Multi-system exposure

5. Major Attack Classes and Mitigation

5.1.Identity and Access Attacks

Identity compromise, OAuth abuse, privilege creep, IAM escalation, and agent identity abuse exploit the permissions surrounding AI systems. Mitigation should include phishing-resistant authentication, short-lived and narrowly scoped tokens, workload identities, privileged-access management, conditional access, periodic entitlement reviews, and independent authorization for high-impact actions. An agent should never inherit broad permissions merely because a user can access the agent.

5.2.Input and Context Manipulation

Prompt injection, malicious documents, and email-to-agent attacks exploit the fact that AI systems process both instructions and untrusted content.

Controls should separate trusted instructions from external data, label untrusted context, sanitize documents, scan email attachments, validate sources, and prevent external content from becoming authorization. Prompt-injection defenses should be layered with tool-level authorization because detection alone cannot guarantee safety.

5.3.RAG, Vector, Data, and Memory Poisoning

RAG poisoning manipulates enterprise knowledge sources so that future retrievals influence model responses. Vector exposure creates confidentiality and privacy risks when embedding stores are insufficiently isolated. Data poisoning can alter training or fine-tuning outcomes, while memory poisoning can persist malicious information across agent sessions. Provenance, integrity checks, source

authentication, access control, tenant isolation, dataset lineage, memory expiration, and revalidation are therefore essential. Recent research shows that targeted corruption of RAG knowledge bases can produce persistent effects on downstream responses (Zou et al., 2025).

5.4. Model and AI Supply-Chain Attacks

Enterprises increasingly obtain models, datasets, libraries, containers, plugins, embeddings, and inference frameworks from third parties. A compromised artifact may introduce hidden functionality or a backdoor. Organizations should maintain model provenance, cryptographic hashes, trusted registries, reproducible build information, dependency inventories, software and model bills of materials, vulnerability assessment, and behavioral validation. Classical machine-learning supply-chain research demonstrates that apparently normal models can contain hidden malicious behavior (Gu et al., 2017). (Gu et al., 2017; Liu et al., 2018).

5.5. Agent and Tool-Based Attacks

Tool-call abuse, MCP compromise, and excessive agency arise when model outputs are connected directly to enterprise operations. The model should request an action, but a separate policy layer should determine whether that action is authorized. Tool allowlists, sandboxing, schema validation, network segmentation, per-tool identities, rate limits, transaction limits, and human approval for high-impact actions provide stronger containment. The principle is simple: model reasoning must not equal authorization. (Debenedetti et al., 2024; OWASP, 2025).

5.6. Output, Browser, Cloud, Repository, and CI/CD Attacks

AI-generated output must be treated as untrusted input when it reaches downstream applications. HTML, SQL, commands, code, API parameters, and structured data require validation, encoding, schema enforcement, sandboxing, or parameterization. Browser agents require isolated sessions and domain allowlists. AI coding assistants require secret scanning and secure software-development controls, while cloud agents should use short-lived workload credentials. NIST secure-development guidance supports integrating AI-specific security practices

throughout the software-development lifecycle (Booth et al., 2024). (OWASP, 2025; Booth et al., 2024).

6. Proposed Enterprise Ai Security Architecture

Figure 1 presents the proposed defense-in-depth architecture. The design places an AI security gateway between enterprise access channels and AI services, while independent controls protect data, models, agents, tools, cloud resources, and software pipelines. A continuous security and governance plane provides policy enforcement, monitoring, data-loss prevention, adversarial testing, and human oversight.

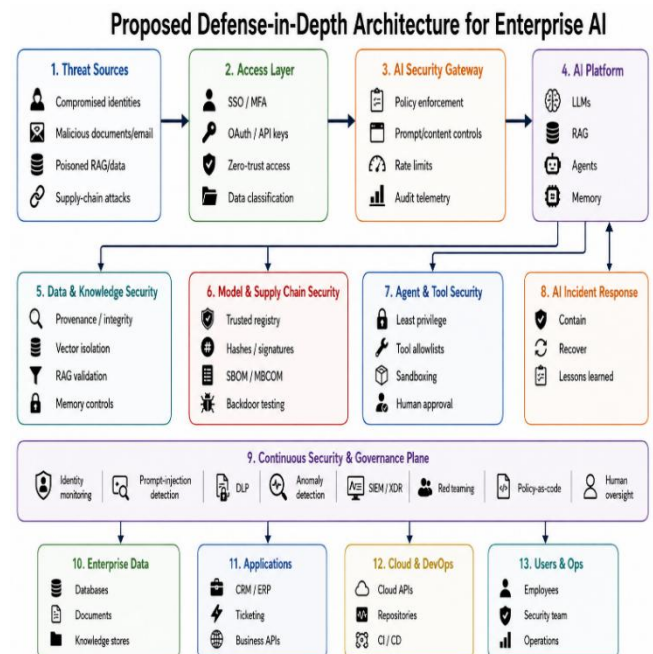


Figure 1 Proposed defense-in-depth architecture for enterprise AI security

The architecture follows a zero-trust principle in which no prompt, retrieved document, model output, agent identity, tool, or external dependency is implicitly trusted. High-impact actions are separated from model reasoning and passed through authorization and policy controls. Security telemetry is collected across the AI lifecycle so that prompt injection, RAG poisoning, credential theft, model compromise, and AI-to-CI/CD pivoting can be detected and contained. This separation reduces the probability that a single compromised component can

directly produce enterprise-wide impact. (Autio et al., 2024; MITRE, 2026).

7. Defense-In-Depth Mitigation Framework

7.1.Governance

Define AI acceptable-use policies, system ownership, risk classifications, data-use restrictions, human-approval requirements, third-party AI requirements, and incident-response responsibilities.

7.2.Identity

Every human, agent, service, model gateway, and tool should have a distinct identity. Implement zero trust, least privilege, continuous verification, and privileged-access controls.

7.3.Data Security

Data entering AI systems should have provenance, classification, integrity protection, access controls, and retention policies.

7.4.Model Security

Treat models as software artifacts and verify origin, hash, version, dependencies, behavior, and training provenance where available.

7.5.Prompt and Context Security

Distinguish system instructions, trusted application instructions, user instructions, and external data. Do not treat these sources as equivalent authorities.

7.6.Agent and Tool Security

Tools should operate through policy enforcement points. Models should never directly perform high-impact operations without authorization.

7.7.Runtime Monitoring

Capture prompt anomalies, tool calls, identity changes, data-access anomalies, model changes, retrieval anomalies, unusual output patterns, and credential use.

7.8.Incident Response

Prepare specifically for prompt injection, RAG poisoning, model compromise, agent credential theft, memory poisoning, unauthorized tool invocation, AI-generated malicious code, and data leakage.

8. Control Matrix

Table 2 is presented as a compact control mapping. Each attack path is paired with a primary preventive control and a secondary detective or containment control.

Identity Compromise — Phishing-resistant MFA;

risk-based access

Oauth Abuse — Short-lived scoped tokens; token monitoring

Privilege Creep — Least privilege; periodic entitlement review

Prompt Injection — Context isolation; runtime detection

Malicious Documents — Sanitization and provenance; content scanning

Email-To-Agent — Action authorization; email trust scoring

Rag Poisoning — Source validation; retrieval monitoring

Vector Exposure — Tenant isolation; encryption

Api-Key Theft — Secret management; rotation

Backend Pivot — Network segmentation; API authorization

Tool-Call Abuse — Tool allowlists; human approval

Mcp Compromise — Tool verification; version control

Memory Poisoning — Memory provenance; expiration policies

Model Supply-Chain Attack — Model signing; SBOM/MBOM

Data Poisoning — Dataset lineage; statistical validation

Model Backdoor — Behavioral testing; model integrity checks

Output Injection — Output validation; sandboxing

Iam Escalation — Privileged workflow approval; policy enforcement

Agent Identity Abuse — Workload identity; credential rotation

Browser Hijacking — Isolated browser sessions; domain allowlists

Ai-To-Ci/Cd Pivot — Secure SDLC; mandatory code review

Repository Leakage — Secret scanning; context minimization

Cloud Credential Theft — Short-lived credentials; behavioral monitoring

Ai Gateway Compromise — Gateway hardening; independent monitoring

9. Results And Discussion

The reviewed attack paths show that the dominant enterprise risk is not a single model vulnerability but the composition of AI with existing identity, data, application, cloud, and software-development systems. Prompt injection is important because it can cross the boundary between data and instructions. RAG poisoning is important because a low-privilege modification to a knowledge source can influence many future interactions. Agent and tool attacks are important because model outputs can acquire operational authority when connected to APIs.

The architecture therefore emphasizes independent authorization and containment. A prompt-injection detector may miss a novel attack, but if tool calls are restricted by capability, resource, identity, and transaction policy, the same missed attack is less likely to cause material damage. Similarly, model-integrity controls may not detect every poisoned data sample, but dataset lineage, behavioral testing, and staged deployment reduce the likelihood of an undetected compromise reaching production. A second finding is that AI security and conventional cybersecurity should be integrated operationally. AI telemetry should feed enterprise SIEM/XDR systems, while identity, endpoint, cloud, application, and data-loss-prevention telemetry should inform AI risk decisions. This creates a unified view of attack chains instead of treating AI as an isolated application. Finally, the review indicates that human oversight should be risk-adaptive. Requiring approval for every low-risk AI action reduces usability, while allowing unrestricted autonomous execution increases risk. A practical design is to classify actions according to data sensitivity, financial impact, privilege level, reversibility, and external exposure, and require human confirmation when the risk crosses a defined threshold.

10. Ai Security Testing and Validation

Traditional penetration testing is insufficient for enterprise AI because the attack surface includes probabilistic behavior and dynamic external context. A mature AI security-testing program should include prompt security testing, agent security testing, model security testing, supply-chain testing, and continuous red teaming. OWASP recommends treating LLM

applications as systems with application, data, and model-specific attack surfaces (OWASP, 2025). (OWASP, 2025; Rando et al., 2024).

Prompt testing should evaluate direct and indirect prompt injection, multi-turn manipulation, prompt leakage, jailbreak attempts, and instruction conflicts. Agent testing should determine whether malicious external content can cause unauthorized tool use, data exfiltration, privilege escalation, financial actions, or configuration changes. Model testing should search for backdoors, poisoning effects, unexpected behavior, memorization, and adversarial weaknesses. Supply-chain testing should verify signatures, dependencies, vulnerabilities, licenses, and behavioral integrity.

11. Ai-Specific Incident Response

AI incident response should extend the conventional prepare, detect, contain, eradicate, recover, and lessons-learned lifecycle. Preparation should maintain an inventory of models, agents, tools, AI APIs, vector databases, knowledge sources, AI gateways, and credentials. During detection and analysis, investigators should determine what component was manipulated, which identity-initiated activity, what context was supplied, which tools were invoked, what data was accessed, and whether the model or retrieval store was modified. Containment may require disabling an agent, revoking credentials, freezing tool access, removing poisoned documents, isolating vector indexes, rolling back a model, or blocking malicious sources. Eradication must remove the root cause. For example, if RAG poisoning occurred, replacing the LLM without cleaning the knowledge base will not resolve the problem. Recovery should include model validation, data validation, credential rotation, re-indexing, security testing, and enhanced monitoring.

12. Research Gaps and Future Directions

Reliable prompt-injection detection remains difficult because malicious instructions can resemble legitimate content. Agent authorization models require new approaches that account for who requested a task, what the agent is allowed to do, why the action is necessary, and what risk it creates. Persistent agent memory needs stronger standards for integrity, expiration, provenance, and recovery after

compromise. AI supply-chain assurance requires standardized provenance and Model Bills of Materials. Automated AI incident response also raises the challenge of using AI to defend AI systems without creating recursive failure modes. (Jin et al., 2024; Shayegani et al., 2023).

Future research should focus on policy-aware AI agents, cryptographic model provenance, secure agent memory, AI-security digital twins, continuous AI red teaming, unified AI security operations, risk-adaptive human oversight, and formal verification of tool permissions. Agents should operate within explicit machine-readable security policies rather than relying solely on natural-language instructions. Security telemetry should become part of SIEM and XDR ecosystems, while human approval should be dynamically triggered according to action risk.

Conclusion

The integration of AI into enterprise environments has created a security landscape substantially broader than conventional application security. AI-powered applications can be attacked through identities, OAuth tokens, prompts, documents, email, retrieval systems, vector databases, models, training data, agent memory, tools, browsers, repositories, cloud credentials, and AI gateways. The 24-path taxonomy demonstrates that these attacks cross identity, context, data, model, agent, application, cloud, and infrastructure layers. The appropriate enterprise response is defense in depth. Organizations should combine zero-trust identity, least privilege, strong authentication, data provenance, secure model supply chains, retrieval security, agent isolation, tool authorization, output validation, continuous monitoring, adversarial testing, and AI-specific incident response. Secure enterprise AI should not assume that models, prompts, retrieved content, tools, identities, or external dependencies are inherently trustworthy; instead, each component should be continuously evaluated and constrained so that a compromise cannot easily become an enterprise-wide incident.

Acknowledgements

The author acknowledges the use of the supplied research-paper template and reference attack-path illustration as formatting and conceptual inputs for

this review. No external financial support is declared for this manuscript.

References

- [1]. Autio, C., Schwartz, R., Dunietz, J., Jain, S., Stanley, M., Tabassi, E., Hall, P., & Roberts, K. (2024). Artificial intelligence risk management framework: Generative artificial intelligence profile (NIST AI 600-1). National Institute of Standards and Technology.
- [2]. Booth, H., Souppaya, M., Vassilev, A., Ogata, M., Stanley, M., & Scarfone, K. (2024). Secure software development practices for generative AI and dual-use foundation models: An SSDF community profile (NIST SP 800-218A). National Institute of Standards and Technology.
- [3]. Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., & Tramèr, F. (2024). AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In Advances in Neural Information Processing Systems.
- [4]. Gu, T., Dolan-Gavitt, B., & Garg, S. (2017). BadNets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733.
- [5]. MITRE. (2026). MITRE ATLAS: Adversarial threat landscape for artificial-intelligence systems. MITRE Corporation.
- [6]. National Institute of Standards and Technology. (2023). Artificial intelligence risk management framework (AI RMF 1.0) (NIST AI 100-1). U.S. Department of Commerce.
- [7]. OWASP. (2025). OWASP Top 10 for LLM applications 2025. Open Worldwide Application Security Project.
- [8]. Zou, W., Geng, R., Wang, B., & Jia, J. (2025). PoisonedRAG: Knowledge corruption attacks to retrieval-augmented generation of large language models. In Proceedings of the 34th USENIX Security Symposium (pp. 3827–3844).
- [9]. Perez, F., & Ribeiro, I. (2022). Ignore previous prompt: Attack techniques for

- language models. arXiv preprint arXiv:2211.09527.
- [10]. Liu, Y., Lee, W.-C., Tao, G., Ma, S., Zhang, Y., & Zhang, X. (2018). Trojaning attack on neural networks. In Network and Distributed System Security Symposium.
 - [11]. Jin, D., Pan, E., Oufattole, N., et al. (2024). What Does the Future Have in Store? A Survey of Prompt Injection Attacks against AI Systems. ACM Computing Surveys.
 - [12]. Wei, J., Haghtalab, N., et al. (2023). Jailbroken: How Does LLM Safety Training Fail? NeurIPS.
 - [13]. Carlini, N., et al. (2021). Extracting Training Data from Large Language Models. USENIX Security Symposium.
 - [14]. Zhang, C., et al. (2024). Prompt Injection Attacks and Defenses in LLM-Integrated Applications. IEEE Access.
 - [15]. Shayegani, E., et al. (2023). Plug-and-Play Prompt Injection Attacks on Large Language Models. arXiv.
 - [16]. Rando, J., et al. (2024). Red Teaming Large Language Models to Reduce Harms. arXiv.
 - [17]. Liu, Y., Lee, W.-C., Tao, G., et al. (2018). Trojaning Attack on Neural Networks. NDSS.
 - [18]. Carlini, N., et al. (2023). Poisoning Web-Scale Training Datasets is Practical. IEEE S&P.
 - [19]. Bubeck, S., et al. (2023). Sparks of Artificial General Intelligence. Microsoft Research.
 - [20]. OpenAI. (2024). GPT-4 System Card. Li, Y., et al. (2024). A Survey of Retrieval-Augmented Generation Security. ACM Computing Surveys.
 - [21]. Wang, Z., et al. (2024). Adversarial Attacks and Defenses for Large Language Models. IEEE Access.
 - [22]. Xu, X., et al. (2025). Security Risks of AI Agents in Enterprise Systems. IEEE Transactions on Dependable and Secure Computing.
 - [23]. Chen, S., et al. (2024). Secure Vector Databases for Enterprise AI Applications. Future Generation Computer Systems.
 - [23]. Sharma, A., et al. (2025). AI Supply Chain

Security: Challenges and Mitigation Strategies. Journal of Information Security and Applications.